U.S.P.S. Express Mail Label No.: EV 329160713US

Date of Deposit: October 17, 2003

Attorney Docket No. 14102US02

SYSTEM AND METHOD FOR RECEIVE QUEUE PROVISIONING

RELATED APPLICATIONS

[01] This application makes reference to, claims priority to and claims benefit from United States Patent Application Serial No. 60/419,354, entitled "System and Method for Statistical Provisioning" and filed on October 18, 2002; United States Patent Application Serial No. 60/420,901, entitled "System and Method for Statistical Provisioning" and filed on October 24, 2002; United States Patent Application Serial No. 60/439,951, entitled "System and Method for Statistical Provisioning" and filed on January 14, 2003; and United States Patent Application Serial No. 60/442,360, entitled "System and Method for Statistical Provisioning" and filed on January 24, 2003.

[02] This application also makes reference to United States Patent Application Serial No. __/___,___ (Attorney Docket No. 14239US02), entitled "System and Method for Receive Queue Provisioning" and filed on October 17, 2003.

INCORPORATION BY REFERENCE

[03] The above-identified United States patent applications are hereby incorporated herein by reference in their entirety.

FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

[04] [Not Applicable]

[MICROFICHE/COPYRIGHT REFERENCE]

[05] [Not Applicable]

BACKGROUND OF THE INVENTION

[06] Conventional systems tend to waste precious resources when communicating via a queue pair (QP) connection. A conventional QP includes a dedicated send queue (SQ) and a dedicated receive queue (RQ) as well as a dedicated or shared completion queue (CQ). Thus, two-way communications can be achieved between peer systems via respective QPs. For example, a message can be sent from an SQ of a first peer system to an RQ of a second peer system. Similarly, a message can be sent from an SQ of the second peer system to an RQ of the first peer system.

[07] When the number of connections between the peer systems increases, the number of QPs increases. Some conventional peer systems allocate the same amount of resources of a preset size to support each QP including the respective dedicated SQ and the respective dedicated RQ. However, such over-provisioning may be wasteful, in particular, with respect to the dedicated RQs. For example, if a particular connection is inactive or relatively inactive, a substantial portion of the resources allocated to the RQ of the inactive or relatively inactive QP remains unused and is not re-allocated for other purposes.

[08] RQ resources are provisioned in anticipation of a burst of messages at wire speed. The user of a transport layer (e.g., a transmission control protocol (TCP) layer) provides a wide-open receive Window to improve performance beyond, for example, 64 KB per connection, thereby forcing the upper layer protocol (ULP) layer to manage the burst situation. Many applications such as Internet Small Computer System Interface (iSCSI) applications do not have a good upper bound as to the number of unsolicited messages that can be sent by one peer and received by another. This may cause connections to be torn down. Managing RQ resources on a per QP basis implies that each QP has to have enough buffers allocated to its RQ to handle a potential burst. When a particular machine serves a high number of QPs, a lot of buffers are allocated to QPs that may utilize them at a very low rate.

[09] Some conventional systems do not control which connections consume buffers and which connections are dropped. In layered protocol processing, when resources are scarce,

data is buffered by a central entity such as a TCP layer until a ULP application procures additional buffering. In rototilled protocol processing, when resources are scarce, data or the connection is dropped requiring either resending of the data or re-establishing the connection. When the ULP application procures additional buffers, then the ULP application may again receive data. In either processing scheme, buffers are consumed on a first-come-first-serve basis. Thus, typically inactive or relatively inactive connections can consume a substantial portion of the available resources, thereby limiting the available resources for typically active connections. Such typically active connections, which tend to be the more important consumers, suffer from reduced traffic bandwidth, are torn down, or have higher costs associated with them for the same level of functionality, for example, as larger memories are provided.

[10] Some conventional systems suffer from substantial delays in adjusting the amount of allocated resources for a particular connection. For example, in a credit-based system at least one round-trip delay between peer systems may be required before a particular resource allocation can be modified. Thus, if a connection suddenly becomes active thereby needing additional resources, then one peer system informs the other peer system which, in turn, acknowledges the change in resource allocation to the original peer system before communications using the adjusted resource allocation may commence. The round-trip delay can be substantial between distant peers or over a network with substantial latencies and may require additional attention from a software layer adapted to provide buffers. In many systems, the response time of such a software entity is large in comparison with network speeds and the response time required by high performance applications. In particular, the round-trip delay becomes compounded each time the resource allocation of the connection is adjusted. The round-trip delays also exacerbate synchronization issues between hardware and software. Moreover, as the number of connections increases, the inefficiencies in adjusting the resource allocations become accentuated.

[11] Further limitations and disadvantages of conventional and traditional approaches will become apparent to one of ordinary skill in the art through comparison of such systems with

some aspects of the present invention as set forth in the remainder of the present application with reference to the drawings.

BRIEF SUMMARY OF THE INVENTION

[12] Some aspects of the present invention may be found, for example, in systems and methods that provide receive queue (RQ) provisioning. In one embodiment, the present invention may provide a communications system that may include, for example, a first queue pair (QP), a second QP and a general pool. The first QP may be associated with a first connection and may include, for example, a first send queue (SQ). The second QP may be associated with a second connection and may include, for example, a second SQ. The general pool may include, for example, a shared receive queue (SRQ) that may be shared, for example, by the first QP and the second QP.

[13] In another embodiment, the present invention may provide a communications system that may include, for example, a network interface card interface (NI) and a consumer. The consumer may be coupled to the NI. The NI may include, for example, a network interface card (NIC) and a NIC driver. The NIC may be coupled to the NIC driver. The NIC may include, for example, a first QP, a second QP and an SRQ in which the first QP and the second QP share the SRQ.

[14] In yet another embodiment, the present invention may provide a method for communicating. The method may include, for example, one or more of the following: establishing a first connection associated with a first QP; establishing a second connection associated with a second QP; concurrently sharing a single receive queue (RQ) between the first QP and the second QP; and provisioning the single RQ using statistical information.

[15] These and other features and advantages of the present invention may be appreciated from a review of the following detailed description of the present invention, along with the accompanying figures in which like reference numerals refer to like parts throughout.

BRIEF DESCRIPTION OF THE DRAWINGS

[16] FIG. 1 shows a block diagram illustrating an embodiment of a system that provides receive queue provisioning according to the present invention.

[17] FIG. 2 shows a representation illustrating an embodiment of a node according to the present invention.

[18] FIG. 3 shows a representation illustrating another embodiment of a node according to the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[19] Some aspects of the present invention may be found, for example, in systems and methods that provide receive queue (RQ) provisioning. In some embodiments, the present invention may provide a system that includes, for example, a shared receive queue (SRQ) that may be shared among a plurality of connections. The resources of the SRQ may be dynamically allocated based upon, for example, statistical information and information (e.g., actual demand information) about the connections.

[20] In some embodiments, the present invention may find use with, for example, remote direct memory access (RDMA) applications or RDMA-based applications such as, for example, Internet small computer system interface (iSCSI) over RDMA (iSER) applications. In addition, the present invention may find use with, for example, kernel mode applications or user mode applications.

[21] In some embodiments, the present invention may provide a controller (e.g., a storage controller) that may be adapted to dynamically assign resources for messages (e.g., unsolicited messages) while supporting a large number of connections. In other embodiments, the present invention may provide a controller that may be adapted to dynamically assign resources per activity on a particular connection. In still other embodiments, the present invention may provide a controller that may be adapted to recoup buffers (e.g., RQ buffers) that have been posted for a particular connection that is becoming or has become inactive. In yet other embodiments, the present invention may provide a controller that may be adapted to enhance efficiency or to match application needs by running in, for example, a kernel mode or a user mode.

[22] FIG. 1 shows a block diagram illustrating an embodiment of a system that provides receive queue provisioning according to the present invention. A first node 10 may be coupled to a second node 20 over one or more connections via a network 30. A particular node may be coupled to one or more other nodes (not shown) via the network 30. The first node 10 may include, for example, a consumer 40, a verb interface 50 and an RDMA-enabled network interface card interface (RI) 60. The verb interface 50 may be include or

may be replaced by an application programming interface (API). The second node 20 may include, for example, a consumer 70, a verb interface 80 and an RI 90. Although some embodiments according to the present invention may employ certain aspects of an RDMA-enabled system, the present invention need not be so limited. Accordingly, some embodiments according to the present invention may employ, for example, a network interface card interface instead of an RI.

[23] The consumer 40 of the first node 10 may be in communications with the consumer 70 of the second node 20. The consumer 40 may be coupled to the RI 60 via the verb interface 50. The RI 60 may be coupled to the RI 90 via the network. The RI 90 may be coupled to the consumer 70 via the verb interface 80.

[24] In operation according to one embodiment, the consumer 40 may communicate with the consumer 70 resulting in, for example, the sending of information from the first node 10 to the second node 20. The consumer 40 may communicate with the RI 60 by using, for example, verbs. The RI 60 may then perform the requested operation. In one example, the requested operation may include, for example, sending and storing data from the first node 10 to the second node 20. Accordingly, data may be sent from the RI 60 to the RI 90 via the network 30. The consumers 40, 70 may be, for example, in two-way communications sending information back and forth via the network 30.

In one embodiment, the present invention may provide that communications are achieved via one or more queue pairs (QPs) in the first node 10 and one or more QPs in the second node 20. A QP in the first node 10 and a QP in the second node 20 may support, for example, a two-way connection between the first node 10 and the second node 20. A plurality of connections may be supported, for example, by a plurality of QPs in the first node and a corresponding plurality of QPs in the second node. Each QP may include, for example, a send queue (SQ) and a receive queue (RQ). In one example, messages may be sent from an SQ in one node and received in an RQ in another node. The node receiving a message may post, for example, buffers in its RQ that may be consumed by the received messages.

[25] In a particular node, a plurality of connections may be supported by a plurality of QPs. Although an SQ of each QP may be dedicated to each connection, an RQ may be shared by one or more QPs (i.e., by one or more connections). Thus, one or more shared RQs (SRQs) may be provided in a general pool of a particular node, thereby providing resources (e.g., buffers) for a plurality of connections.

[26] In some embodiments according to the present invention, the use of an SRQ by one node may not be known or visible to another node. The SRQ may be, for example, a local concept or a local policy. RQ resources corresponding to communications with multiple remote nodes may be locally grouped and provisioned by one or more SRQs. In one embodiment, multiple remote nodes may be served by one or more SRQs. In another embodiment, a single remote node with multiple connections may be served by one or more SRQs. In yet another embodiment, a group of nodes associated with a particular application or with a particular trust level may be served by one or more SRQs.

[27] Each connection maintained by the nodes 10, 20 may have its own buffering needs which may change with time and which may depend upon the type of connection. Thus, the nodes 10, 20 may provide statistical provisioning of resources (e.g., buffers) for the connections.

[28] In one embodiment, the present invention may provide that the nodes 10, 20 provide statistical provisioning for the SRQ resources. Thus, a node 10, 20 may provide local control for the provisioning of a local SRQ. The use of a local SRQ by one node may not necessarily be known or visible to another node. Local control may minimize delays, for example, in adjusting resource allocations. In addition, local control may support, for example, real-time resource management. The information used in determining the statistical provisioning may be include, for example, static data (e.g., data tables programmed with preset values) or dynamic data (e.g., local empirical data which may change over time). Furthermore, other information that may affect statistical provisioning may include, for example, connection parameters such as, for example, the type of connection, the number of connections and the status of connections.

[29] In another embodiment, the present invention may provide that the nodes 10, 20 provide local control over the resource allocation of each connection. For example, a node 10, 20 may provide a limit as to the number of buffers (e.g., buffers of the SRQ) a particular connection may access at a given time. Thus, for example, the node 10, 20 may make more buffers of an SRQ accessible to an active connection (e.g., a storage target connection) than to an inactive or relatively less active connection (e.g., a desktop connection for e-mail which is accessed only a few times daily). Furthermore, as active connections become less active, the node 10, 20 may make fewer buffers accessible to such less active connections. In the event that a particular connection limit is exceeded, the node 10, 20 may, for example, increase the limit, replenish the buffers, drop the connection or drop data packets carried by the connection. The node 10, 20 may also, upon exceeding a limit, analyze the connection behavior to determine whether, for example, to retune resources, adjust the limit or dry out resources if the connection behavior is indicative of a malicious attack. In one example, if a desktop e-mail connection (i.e., a relatively inactive connection) suddenly becomes active, thereby exceeding its local limit, then the desktop connection may be dropped. Accordingly, a less important connection may be dropped before consuming too many buffers, thereby possibly taking away resources from more important connections (e.g., a database connection that is generally very active).

[30] Limits may be, for example, soft or hard. A soft limit may generate, for example, a local indication to the local buffer management that can analyze connection behavior to determine a course of action. The soft limit may provide a graceful manner in which to handle error conditions or unforeseen conditions (e.g., bursty conditions). A hard limit may trigger an automatic response (e.g., closing a connection) as determined by the local buffer management.

[31] To respond to out-of-order messages (e.g., which may be a form of malicious attack), a QP may consume buffers for the messages in between the next in-order message M_{N+1} and the out-of-order message M_{N+x}. This may guarantee that the QP has enough resources for all messages and may simplify operation. The QP may use the sequence number of the incoming message (e.g., as provided by a direct data placement (DDP) protocol message) to

determine the number of expected messages between the next in-order expected message M_{N+1} and the out-of-order message M_{N+x} carrying a sequence number of N+x. The QP may then be allocated buffers for all of the messages (i.e., x buffers). If x is large, limited by the only the TCP receive window, then it may starve other QPs. In one embodiment, the buffer manager may impose an out-of-order limit (e.g., setting the out-of-order limit to x) on each QP to prevent it from a particular QP from starving other QPs as may occur during a malicious attack. The out-of-order limit may be different from other limits in other layers dealing with out-of-order messages. In another embodiment, the buffer manager limit may be combined with other limits such as, for example, a TCP limit on the number of holes (i.e., one or more ranges in sequence space between the sequence numbers of received messages) that can be managed. If either of the limits are exceeded, then a frame may be dropped. In providing buffers to the SRQ, the buffer manager may take into account that each QP may have out-of-order messages.

[32] FIG. 2 shows a representation illustrating an embodiment of a node according to the present invention. The node may include, for example, a consumer 100, a verb interface 110 and an RI 120. The consumer 100 and the RI 120 may be in communications with each other via, for example, the verb interface 110. The RI 120 may include, for example, an RDMA-enabled network interface card (RNIC) 130 and an RNIC driver and library 140. The RNIC 130 may include, for example, a data engine layer 150 and a protocol processing layer 160. The RI 120 may also include, for example, a plurality of QPs (e.g., QP1 and QP2) and a general pool. In one example, each QP may include, for example, an SQ, a limit receive queue (LRQ) and an RDMA read request queue. The general pool may include, for example, a shared completion queue (SCQ), an SRQ and a memory translation and protection table (TPT) for the resources attached to the SRQ. The SCQ and the SRQ may be shared by at least some of the QPs. However, a QP that shares an SRQ need not necessarily share an SCQ. There need not be a link between QPs sharing one or more SRQs and QPs sharing one or more SCQs.

[33] FIG. 3 shows a representation illustrating another embodiment of a node according to the present invention. The node illustrated in FIG. 3 has many of the same components as

the node illustrated in FIG. 2. Referring to FIG. 3, a buffer manager 170 may manage, for example, the general pool including, for example, the SRQ. The buffer manager 170 may also manage the LRQs of the QPs. Although illustrated as a single buffer manager 170, a plurality of buffer managers may be used. Also, although illustrated as part of the consumer 100, the buffer manager 170 may be part of other components or shared by other components such as, for example, the RI 120. FIG. 3 shows that the present invention also contemplates that the CQ may be a dedicated CQ disposed within each QP.

[34] Referring to FIG. 2 or FIG. 3, the consumer 100 may include, for example, a verb consumer and may be in the form of, for example, a software program or an application. The consumer 100 may be, for example, a kernel mode application or a user mode application. In one embodiment, the consumer 100 may be, for example, an application (e.g., a storage application) that is trusted by the operating system of the node. The consumer 100 may even be part of the operating system. In another embodiment, the consumer 100 may be, for example, an application (e.g., a third party application) that need not be entirely trusted by the operating system of the node. The consumer 100 might not even be part of the operating system and may be deemed to be a relatively unsecured application.

[35] The verb interface 110 may provide an interface (e.g., a software process) between the consumer 100 and the RI 120. Communications between the consumer 100 and the RI 120 may be achieved, for example, through the use of verbs which may perform, for example, one or more of the following: managing a connection state; managing the memory TPT; submitting work to the RNIC 130; and retrieving work and events from the RNIC 130.

[36] The RI 120 may perform work, for example, on behalf of the consumer 100. The RI 120 may include, for example, software, firmware, hardware or combinations thereof. The RI 120 may be adapted to perform, for example, one or more of the following: queuing; managing memory; converting work requests (WRs) into work queue elements (WQEs); supporting RDMA, DDP or other protocols; converting completion queue elements (CQEs) into work completions (WCs); and generating asynchronous events.

[37] Work may be submitted in the form of a WR. The WR may include, for example, a scatter/gather list (SGL) of local data segments. A local data segment may include, for example, a local steering tag (STag), a local target offset, a length or other modifiers or descriptors. Some examples of WR types include, for example, send, RDMA write, RDMA read, bind memory window, etc.

[38] The RI 120 may convert WRs into WQEs and may process the WQEs. The RI 120 may return control to the consumer 100 after the WR has been converted to a WQE and submitted to the SQ. In one embodiment, the RI 120 may not modify the WR after control has been returned to the consumer.

[39] Tagged buffers may be used, for example, for solicited activity. Tagged activity may employ, for example, the SQ. Untagged buffers may be used, for example, on the SRQ. The CQE may be used, for example, to report completion for the posted SQ/SRQ WQEs. The CQ may be shared or non-shared. Statistical provisioning of the CQEs may be used. In some embodiments, some minimum may be guaranteed per a verb specification to ensure proper operation.

[40] The protocol processing layer 160 may process some of the layered protocols encapsulated in incoming packets using, for example, layered or rototilled protocol processing. In one example, the protocol processing layer 160 may process, for example, one or more of the following protocols: an upper layer protocol; a marker-based upper layer protocol data unit (ULPDU) aligned (MPA) framing protocol or a framing protocol data unit (FPDU) protocol; a transport protocol (e.g., a transmission control protocol (TCP)); an Internet Protocol (IP); and a data link layer protocol (e.g., an Ethernet). Examples of some MPA framing protocols may be found in, for example, United States Patent Application Serial No. 10/230,643, entitled "System and Method for Identifying Upper Layer Protocol Message Boundaries" and filed on August 29, 2002. The above-referenced United States patent application is hereby incorporated herein by reference in its entirety. Other examples of some MPA framing protocols may be found, for example, in conventional MPA framing protocols.

[41] With regard to RQ sharing, one or more of the following may be applicable according to the present invention. RQ may be a central resource. A buffer manager such as, for example, one or more buffer managers 170, may allocate limits to all QP. Each QP may draw based upon its local limit. The top of the CQ processing may determine credit consumption by the QP. New credit may be replenished to each QP on its credit receive queue (CRQ).

[42] In one embodiment, the system may support SRQ resources less than, for example, a total limit. A local limit may be set by the LRQ for each QP, thereby setting a local maximum consumption for a respective QP. The local limit as well as total limit may be subject to statistical provisioning. If a local limit has been reached without replenishment, then tear down may occur on the overdraft.

[43] One or more embodiments according to the present invention may include, for example, one or more advantages as set forth below.

[44] A general pool implementation may be applicable to either kernel space or user space. The SRQ may be used as a general pool for use with, for example, receive-untagged WQEs. Such an implementation may have, for example, one or more of the following advantages. Resources may be administered locally and may be more flexible. The general pool need not necessitate a flow control for ULPs (e.g., iSER) using the RI nor changes to the ULP or to the wire protocols. A round trip delay might not be assessed each time a resource allocation is adjusted. The general pool implementation may provide scalability for a large number of QPs. The implementation may provide a simplified management scheme of buffers for use with, for example, unsolicited activities, commands and task management. Verbs used with respect to sharing the CQ may also be applicable to sharing the RQ. WQEs need not be completed in order to the CQ. Some embodiments may include buffers, which are all the same size, and may share regions among participating QPs. As the limit may be optional as well as on top of the SRQ, in some embodiments according to the present invention, latencies may not be present as the QPs may manage the buffer consumption.

[45] In some embodiments, the present invention may provide an SRQ for use, for example, in a user mode (e.g., an RDMA user mode). Some applications (e.g., database applications, interprocess communication (IPC) applications, etc.) may concurrently use a plurality of QPs. For example, on a particular Tier 2 or a particular Tier 3 server, the number of multiple concurrent QPs may range, for example, from approximately 1,000 QPs to approximately 64,000 QPs. By using an SRQ, instead of each QP having its own solely dedicated RQ, precious system resources may be conserved. For example, if a particular connection is relatively inactive, then statistical provisioning may allocate less than the respective maximum amount of resources to the relatively inactive connection. As a numerical example, instead of providing a full window of 64 KB for every connection of 1000 connections for a total of 64 MB, it may suffice to provide for the longest burst at wire speed before the buffer manager can react by providing more resources (e.g., for a long 1 ms latency of the buffer manager on a 1 gigabit network only 1 MB may be required). In another example, a shared pool may save a substantial amount of system resources, for example, at least for threads of the same process where security schemes (e.g., protection domain schemes) allow.

[46] The RNIC 130, which may have limited on-chip memory, may be adapted to keep state for all active QPs. This may include, for example, a few entries of the SRQ (e.g., a few WQEs). The RNIC 130 may be adapted to handle, for example, out-of-order reception (e.g., potentially receiving message M_{N+x} into WQE_{N+x}), without substantially increasing on-chip caching requirements. In one embodiment, on-chip caching requirements may be alleviated by employing an SRQ in a general pool that may result in less context being kept by each QP. Savings due to the general pool configuration can be quite substantial. In one example, savings of up to approximately 20 percent may be realized. Additional savings including other savings values in excess of approximately 20 percent are also contemplated by the present invention.

[47] The scalability of the RNIC 130 may be enhanced by caching some of the QP context. QP context may be fetched when a respective QP becomes active or becomes increasingly active. Caching may be further enhanced by reducing the size of the QP

context. In one example, SRQ WQEs per QP might not be stored. One or more embodiments may benefit from fewer bytes being transferred over a particular time period (e.g., a limited time). Furthermore, an improved real-time solution may be provided with less host bus (e.g., PCI, PCI-X or PCI-Express) arbitration and fewer host bus transactions.

[48] While the present invention has been described with reference to certain embodiments, it will be understood by those skilled in the art that various changes may be made and equivalents may be substituted without departing from the scope of the present invention. In addition, many modifications may be made to adapt a particular situation or material to the teachings of the present invention without departing from its scope. Therefore, it is intended that the present invention not be limited to the particular embodiments disclosed, but that the present invention will include all embodiments falling within the scope of the appended claims.